

Java 기초

2. 자바언어의 특징

2022-06 백성애

1. Java 개발 Platform

Java SE (Standard Edition)

- 데스크탑, 서버, 임베디드 시스템 개발을 위한 플랫폼 (Core and Desktop)

Java EE (Enterprise Edition)


- WAS의 지원으로 실행되는 시스템 개발을 위한 플랫폼 (Servlet/JSP/EJB...)

Java ME (Micro Edition)

- 모바일 장치나 내장형 장치에서 실행되는 애플리케이션 지원 플랫폼 (Embedded)



2. Java의 역사

- 1) 1991년 sun사 Green Project 출범 -James Gosling을 주축으로 Oak라는 언어 개발
[가전기기에서 사용할 목적] 하드웨어 독립적인 언어로 구상 됨
 - 2) 1995년 : sun사와 netscape사 협약
 - 3) 1996년 : 자바 지원 netscape 2.0 발표
 - 4) 1997년 : jdk1.1 발표
 - 5) 1998년 : jdk1.2 발표
 - 6) 2000년 : jdk1.3 발표
 - 7) ~ : jdk1.4/ 5.0 / 6.0/ 7.0/8.0버전 발표
 - 8) 2022년 현재: jdk18버전
- 

3. Java 의 특징

- 1) 플랫폼 독립성 : JVM(Java Virtual Machine)이 해당 플랫폼마다 제공되어져, 이를 설치하면 어떤 운영체제에서 작성된 자바 파일이든지 동일한 실행을 제공한다.

Write One Run Anywhere

- 2) 객체 지향언어 : 재사용성, 유연성, 프로그램 생산성 향상



3. Java 의 특징

- 3) 멀티 스레드 지원 : Thread는 Process보다 작은 단위로 동시 다발적으로
작업 수행이 가능.
- 4) 자동 메모리 관리
 - Garbage Collector(쓰레기 수집기)
- 5) 동적인 성능 확장 제공



4. 자바의 주석 처리 방법

1) // : 단문 주석

2) /* */ : 복문 주석

3) /** */ 문서화 주석

-javadoc를 이용해서 **API**문서를

작성하고자 할 때 사용



5. 클래스의 구조

1) 패키지 선언: **최상단에 위치**

`import`문 보다는 먼저 와야 한다.

2) `import` 문 : 사용하고자 하는 패키지 경로를 기재

3) `class` 선언 : `class` 키워드로 선언하고 클래스 이름을 기재.

이 때 주의. **클래스명==파일명**



6. 클래스(class)의 멤버 - 변수(Variable)

1) 변수 : 데이터를 임시적으로 저장하는 메모리 공간.

즉 변수란 값을 저장하는 메모리 공간의 위치를 의미.

2) 변수의 종류

└) 멤버변수(instance 변수)


ex) int a=10;

**객체명으로 접근해야 한다.

└) 클래스변수(static 변수)

ex) static int b=10;

**클래스명으로 접근해야 한다.



6. 클래스(class)의 멤버- 생성자(Constructor)

2) 생성자(멤버 변수의 초기화): 객체를 생성할 때 호출된다.

생성자 이름과 클래스 이름은 같아야 한다.

반환타입이 없다.

ex)

```
class Hello{  
  
    public Hello(){  
  
        a=20;  
  
    //생성자  
  
    }
```

6. 클래스(class)의 멤버- 메소드(Method)

1) `public static void main(String args[]){ }`

: 실행시 제일 먼저 JVM에서 호출해주는 메소드.

프로그램 시작이자 끝이 된다.

2) 사용자 정의 메소드

```
public int myFunction(int a){  
    메소드가 하는 일  
    this.a=a;  
    return a;  
}
```

7. 변수의 명명규칙

- 영문자와 숫자를 섞어 쓸 수 있으나, 숫자로 시작되어선 안된다.
- 한글/한자도 변수명으로 사용가능
- 특수문자는 변수로 사용할 수 없다.

단, 언더바(_), \$는 식별자로 사용 가능

- 변수명은 명사형으로 지으며, 소문자로 시작.

keyword는 사용 불가

****keyword->**뒷페이지 참조



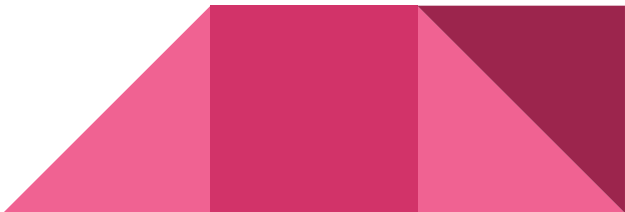
Java keyword

abstract	continue	for	new	switch
assert	default	goto	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const	float	native	super	while

※ 잘못된 변수 선언의 예

- int 9nine : 숫자로 시작 불가
- int hey&bar: &라는 특수문자 사용 불가
- int char : 예약어는 사용 불가

다음변수는?

- int 변수=10;
 - int \$\$\$=20;
 - int _myVar=30; ---> 모두 사용 가능.
- 

8. 자바의 자료형

- 1) Primitive Type : 기본 자료형
- 2) Reference Type : 참조형 ex) String s="Hi";
String s=new String("Hi");
 - +---ㄱ) 클래스형
 - +---ㄴ) 인터페이스형
 - +---ㄷ) 배열

1) Primitive Type

- + ㄱ) 수치형--정수형----byte
 - +-----short
 - +-----int
 - +-----long
- 실수형
 - +-----float
 - +-----double
- +ㄴ) 문자형 - char : '가' 'A' '\u0000'
0~ 65535[16비트]
- +ㄷ) 논리형 - boolean : true, false

9. 자바의 연산자 종류

1) 분리자 : . [] () ; ,

2) 단항 연산자: 항이 하나인 연산자

ㄱ) 증감연산자 : ++ --

ㄴ) 부호연산자 : + -

ㄷ) 비트별 NOT 연산자 : ~

ㄹ) 논리 부정 연산자 : !

3) 산술 연산자 : * / % + -

4) 쉬프트 연산자 : << >> >>>

5) 비교 연산자 : < <= > >= instanceof

6) 비트 연산자 : & ^ |

7) 논리 연산자 : && ||

8) 조건 연산자 : ? :

9) 할당 연산자 : = += *= /= -=
<<= >>= >>>= &= ^= |=

10. 자바의 제어문

주 제어문	보조 제어문
<p>1) 조건문 :</p> <ul style="list-style-type: none">if, if~else,if~else if~else <p>2) switch~case문</p> <p>3) 반복문</p> <ul style="list-style-type: none">- for문- while문- do~while문	<p>1) break 문</p> <p>2) continue 문</p> <p>단독으로 쓰이지는 못하고 주제어문과 함께 사용된다.</p>

11. Wrapper 클래스

기본자료형을 마치 랩으로
포장해놓은 것
같다하여 래퍼 클래스라고 함.
기본자료형은 단순한 연산에
사용되지만 래퍼 클래스는
참조형이므로 변수와
다양한 메소드등을 가져 많은 기능을 수행한다.

Primitive Type(기본자료형)	Reference type(참조형)
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
char	Character
boolean	Boolean

12. 배열(Array)

1] 배열이란? ..동종의 데이터들을 묶어 저장해놓은 자료구조

- 비슷한 구조의 것을 하나의 데이터 구조에 번호를 매겨 저장하는 방식을 의미.
- 이런 방법은 데이터의 저장, 정렬, 검색을 매우 유용하게 할 수 있어 편리하다.

[2] 배열 사용 방법

1) 선언

2) 메모리 할당

3) 초기화



12. 배열(Array)

-1차원 배열

데이터형 배열명[] = new 데이터형[배열의 크기];

-2차원 배열


데이터형 배열명[][] = new 데이터형[배열의 크기][배열의 크기];

예1) int a[]; //1) 배열 선언

 a = new int[2]; //2) 메모리 할당

 a[0] = 10; //3) 초기화

 a[1] = 20;



12. 배열(Array)

예2) 선언과 메모리 할당을 동시에 하는 방법

```
int b[]=new int[3]; //1)+2)
```

```
b[0]=100; //3) 초기화
```

```
b[1]=200;
```

```
b[2]=300;
```

```
b[3]=400; [x] //배열 index초과 오류 발생
```

예3) 선언,메모리할당, 초기화를 한꺼번에 하는 방법

```
int [] c={1,2,3,4,5};
```



12. 배열(Array)

****배열에 저장된 값을 꺼내고자 한다면...**

그때는 **index**를 이용해 꺼내온다.

index는 0부터 시작.

이때 주의. 인덱스가 배열 크기를 벗어나지
않도록 주의.

예) `System.out.println(a[0]);`



12. 배열(Array)

[3] 다차원[-2차원] 배열 사용 방법

1) 선언

```
int arr[][];
```

```
int [][] arr; int []arr[];
```

2) 메모리 할당-배열 생성

```
arr=new int[3][2]//3행 2열
```

3) 초기화

```
arr[0][0]=1; arr[0][1]=2; arr[1][0]=3;
```

```
arr[1][1]=4; arr[2][0]=5; arr[2][1]=6;
```



12. 배열(Array)

예1) 선언과 동시에 생성하고 초기화

```
int arr[ ][ ]={ {1,2,3},{10,20,30} };
```

예2) 이차원 배열의 경우, 배열을 생성할 때 행의 크기를 고정시키고, 각 행에 대한 열의 크기를 가변적으로 줄 수 있다.

```
int []arr[]=new int[3][ ];
```

//행의 크기를 3으로 고정. 열의 크기는 나중에 할당.

****열의 크기 할당 방법****

```
arr[0]=new int[2];
```

```
arr[1]=new int[1];
```

```
arr[2]=new int[4];
```



12. 배열(Array)

```
arr-----> +-----+  
              |arr[0][0] | arr[0][1]|  
              +-----+  
              |arr[1][0] |  
              +-----+  
              |arr[2][0] | arr[2][1] |arr[2][2] | arr[2][3]|  
              +-----+
```

```
arr----->| arr[0] | arr[1] | arr[2] |  
           ||  
           |  
           ▽  
           |arr[0][0]|arr[0][1] |
```